Backup Policy and Backup

Group-5: Team-1

Jay Sharma - **2018101033**
Tanmay Garg - **2018102021**
Vedant Mundheda - **2018112006**

Aim

- Backup and recover datasets stored on the system.
- Maintain version history to access previous versions of the datasets.
- Store only the modifications to datasets (between successive versions) to keep the size small.
- Be able to handle all file formats and directory structures.
- Easy and intuitive backup and recovery API calls should be available.
- For backups to be practically safe, they should be on different system, meaning backup is not local but via FTP.
- A web based UI for non-programmers to backup and restore their datasets.

Extended Aim

Explore how to backup:

- MySQL databases
- MINIO Objects
- Label-Studio Annotations
- File-type agnostic arbitrary backup

Base Backup Functionality

```
make_backup(backup_path, dataset_path, full=False)
```

- For backing up a dataset, simply call the “make_backup” function.
- This function takes care of incremental backups, maintaining version history, and storing only those files which are needed to save space.

```
recover_backup(backup_path, dataset_path, out_folder, backup_id=None, req_time=None)
```

- For restoring a dataset using its backup, just call the “recover_backup” function.
- This takes care of how to backup from various full/partial backups, as the dataset was before given a time.

Backup Structure

Dataset Folder

```
test
|   file1.txt
|   file2.txt
|
\---lmao
    |   fly.txt
    |
    \---phoenix
            fire.txt
```

Recovered Folder

```
test_recovered
|   file1.txt
|   file2.txt
|
\---lmao
    |   fly.txt
    |
    \---phoenix
            fire.txt
```

Backup Folder

```
test_backup
|   full_backup_indices.txt
|
+---backup_0
|   |   backup_time.txt
|   |   paths.txt
|   |
|   \---data
|       |   file1.txt
|       |   file2.txt
|       |
|       \---lmao
|           |   fly.txt
|           |
|           \---phoenix
|                   fire.txt
|
\---backup_1
    |   added_list.txt
    |   backup_time.txt
    |   deleted_list.txt
    |   modified_list.txt
    |   paths.txt
    |
    +---added
    \---modified
            0
```

Say we have a folder named “test” to be backed up.

When we call make_backup on it, it creates a folder of the specified name inside the backup path. Then, it creates a folder named backup_0, which is a full backup (you can make any subsequent backup as full by passing the parameter full=True). Then if we modify some file (say “fly.txt”) in the dataset, and call make_backup again, it creates a backup_1 which is a partial backup by default.

Then, if we call restore_backup, with output folder as “test_recovered”, then we get the latest contents of the dataset back.

The directory trees of the dataset “test”, the backup folder, and the recovered dataset folder “test_recovered” are given here.

MySQL Database Backup

- MySQL databases are stored in form of IBD (InnoDB engine files) by default. [Database folders containing an IBD file for each table].
- If you lose these files due to corruption, they can be restored. However, the database can't be restored DROP TABLE, TRUNCATE TABLE, or DROP DATABASE commands are used.
- Backup the IBD file folder using the make_backup API call. Then, to restore, call recover_backup to the MySQL data folder (default location is "C:\ProgramData\MySQL\MySQL Server 8.0\Data").
- To make MySQL recognise those files, open the MySQL shell,
    - type "USE db_name", and for each table type:
    - ALTER TABLE table_name DISCARD TABLESPACE
    - ALTER TABLE table_name IMPORT TABLESPACE

REF: https://dev.mysql.com/doc/mysql-enterprise-backup/3.11/en/partial.restoring.single.html

MINIO Object Backup

- If the data is stored in a distributed manner, it would be tough to back it up with this, consider using backup utilities catered to MINIO. [use “mc mirror” command or a utility like Restic/Rclone]
- If the data is only local and data is stored in a MINIO data bucket, then the functions “make_backup” and “recover_backup” can be called at the location where the bucket is stored. (the location is mount_name/bucket_name).
- For example, if you started MINIO server by typing “minio server /tmp”, and created a bucket named “jarvis”, then the data can be found in (and be backed up from) the path “/tmp/jarvis”.

REF:
https://serverfault.com/questions/850998/how-to-keep-minio-backed-up
https://github.com/minio/minio/issues/4135

Label-Studio Annotations Backup

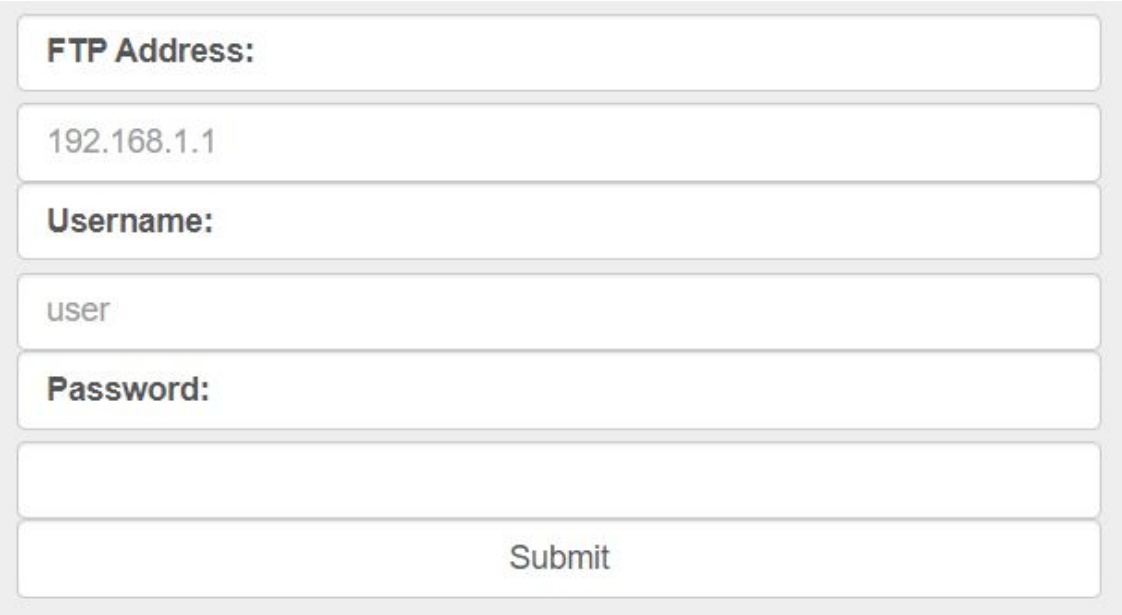
- The same functions “make_backup” and “recover_backup” can be called in whatever location the data is exported to.
- The data can be exported using “Export” feature in Label-Studio. A path is specified to export the data to, same path can be used for backup.
- If the annotations are not complete, use the “Create New Snapshot” feature in Label-Studio. A path is specified to store the snapshot, same path can be used for backup.

REF: https://labelstud.io/guide/export.html

FTP based backup

- Practically speaking, backups should be stored on a separate system, different from the system containing datasets.
- For this, we need to fetch and send data from backup-server to dataset-server while backing up and recovering respectively.
- We wrote the directory-tree comparisons, modified time checks, recursive directory copier for FTP, and rewrote the code for make_backup and recover_backup to accommodate FTP functionality.
- We chose FTP over SSH and HTTP, because it is specifically catered towards file-transfer, providing less overhead. SSH and HTTP can possibly be made to do the same thing, but SSH is better for interactive remote sessions from one system to another, while HTTP is better for web-hosting and non-file related requests.
- One can reuse the FTP functions we made as APIs as well (can be found in ftp_utils.py). It's because we constructed basic low-level functionality like copying directories from scratch using FTP, which someone else might need for their project.

Flask based UI



View Backups View Datasets

Backups View

Recover from this backup

<= Back

- backup_0
- backup_1
- backup_2
- full_backup_indices.txt

View Backups View Datasets

Datasets View

Backup this directory

<= Back

- a
- file1.txt
- file2.txt
- lmao

Backup Path: backup_folder/folder2

Dataset Path:

dataset_folder/folder2

Output Path

dataset_folder/folder2_recovered

Recover state of the dataset before:

2022 4 29

Submit

:Backup Recovery Form

Dataset Backup Form:

Dataset Path: dataset_folder/folder2

Full Backup?

Submit

Contributions

We had an **equal contribution** from each team member. The exact breakdown is given below:

- Jay Sharma
  - API:make_backup
  - MySQL backup
  - Documentation
  - FTP based low level functions
  - Flask server backend
  - Frontend: FTP-server login form
- Tanmay Garg
  - API:recover_backup
  - MINIO backup
  - Exploring third party backup-tools
  - FTP server
  - Frontend backup-form, recovery-form
- Vedant Mundheda
  - Testing API and UI
  - Documentation
  - Label-Studio Backup
  - Frontend: Dataset-view, Backup view

Thank You